Application No. 10/666,316
Reply to Office Action of: May 4, 2007

Amendments to the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

Listing of claims:

1. (currently amended) A version control system for managing multiple versions of the same file being versioned files used in software development, the version control system comprising:

a) a central server adapted to store a repository of at least one set of data for obtaining a plurality of versions of a file said versioned files, said central server also being adapted to control all modifications to said data and to create new versions of said file versioned files;

b) a proxy server connected to said central server, said proxy server including a read-only cache adapted to store copies of currently and previously accessed ones of said plurality of versions [[data]] obtained from said repository through communication with said central server, said proxy server also being adapted to provide a requested version from said read-only cache when available and by requesting said requested version from said central server otherwise; and

c) at least one client connected to said proxy server, said at least one client comprising a version manager adapted to generate requests for a copy of said requested version said versioned files through said proxy server to reduce network traffic between said proxy server and said central server and to reduce load on said central server.

2. (previously presented) A version control system according to claim 1 further comprising a plurality of branches and a plurality of proxy servers; wherein, for a first branch of said plurality of branches, said central server is adapted to store a list of proxy servers selected from said plurality of proxy servers and associated with said first branch; and wherein the version control system is configured to send an update to notify each proxy server in the list when a change is made to said first branch.

3. (previously presented) A version control system according to claim 1, wherein said central server includes an access control system adapted to validate requests received by said central server.

2

4. (cancel)

5. (cancel)

6. (previously presented) A version control system according to claim 1, wherein the at least one client is adapted to modify the repository through said central server.

7. (original) A version control system according to claim 6, wherein the central server includes a checkout mechanism for controlling modification to the repository.

8. (original) A version control system according to claim 7, wherein the central server includes a log of changes made to the repository.

9. (previously presented) A version control system according to claim 8, wherein the log is used to update said proxy server after a disruption to the connection between the proxy server and the central server.

10. (currently amended) A version control system according to claim 1 further comprising
a plurality of chained together proxy servers adapted to serve a geographical area,
        wherein each proxy server of said plurality of chained together proxy servers is
        connected to at least one client, said plurality of chained together proxy servers
        each being connected to one proxy server, said one proxy server being
        connected to said central server; [[and]]
a regional proxy server connected to said central server and to each proxy server of said
        plurality of chained together proxy servers,
wherein updates from the central server to the plurality of chained together proxy servers
are first sent to the one regional proxy server.

11. (currently amended) A method of modifying a repository of multiple versions of the same file versions of files in a version control system used in software development, said version control system including a central server and a first interconnected client interconnected to said central server through a first intervening proxy, a second client, and a second proxy, said

3

method comprising the following steps:

a) the first interconnected client requesting from the central server, through the first intervening proxy, a lock on a <u>requested</u> version of a file in the version control system, said requested version being one of a plurality of versions of said file recoverable from a set of data stored in a repository at said central server;

b) the central server checking whether the <u>requested</u> version is unlocked and, if the <u>requested</u> version is unlocked, granting the request of said first interconnected client through said first intervening proxy;

c) the central server sending an update to said second client and to said second proxy <u>notifying the second client of the granting of the request</u>.

12. (previously presented) A method according to claim 11, wherein said lock permits only said first interconnected client to modify said version of said file.

13. (previously presented) A method according to claim 11, further comprising the step of said first interconnected client performing modification to said version of said file and returning the modification to said central server through said first intervening proxy.

14. (previously presented) A method according to claim 13, further comprising the step of said central server sending said modification to said second client and to said second proxy.

15. (currently amended) A central server in a version control system used in software development <u>configured manage multiple versions of the same file</u>, the version control system including a proxy server having a read-only cache and connecting a client to the central server, the central server comprising the following:

a) a repository of <u>at least one set of data for obtaining a plurality of versions of a file</u> <s>versioned files</s> used in said software development;

b) a version manager adapted to provide a <u>requested</u> version of [[a]] <u>said</u> file from said repository to said proxy server <u>and adapted to control modifications to said data and to create new versions of said file</u>;

c) an access control system adapted to manage a request from the client to modify the <u>contents of said</u> repository; and

d) a memory structure adapted to store a log of modifications made to the <u>contents of</u>

Application No. 10/666,316
Reply to Office Action of: May 4, 2007

said repository and adapted to store a list of portions of said repository also being stored in the read-only cache of the proxy server.

16. (previously presented) A central server according to claim 15, wherein said log is used to update said read-only cache in said proxy server after a disruption to the connection between said proxy server and said central server.

17. (currently amended) A proxy server in a version control system for managing multiple versions of the same file being used in software development, the version control system including a client and a central server containing a repository of at least one set of data for obtaining a plurality of versions of a file versioned files, said proxy server comprising the following:

a) a read-only cache adapted to store copies of said requested version and any currently and previously accessed ones of said plurality of versions obtained a requested version of a file requested by the client and received from said repository through communication with said central server; and

b) a version provider adapted to provide the requested version of the file to said client, the version provider being configured to, upon receiving a request for said requested version from said client, first check the read-only cache for the requested version to provide the requested version from said read-only cache when available and, if the requested version is not available found, to request the requested version from said central server on behalf of said client.

18. (currently amended) A proxy server according to claim 17, wherein the read-only cache is configured to store copies of more than one of said [[a]] plurality of versions of said file [[files]] requested from said central server by said client.

19. (currently amended) A computer-readable medium containing processor instructions for implementing a version control system for managing multiple versions of the same file to be used in software development, including instructions for the following steps:

a) storing a repository of at least one set of data for obtaining a plurality of versions of a file versioned files on a central server configured to control all modifications to said data and to create new versions of said file versioned files;

b) establishing communication between said central server and a proxy server including a read-only cache configured to store <u>copies of currently and previously accessed ones of said plurality of versions</u> [[data]] obtained from said repository through communication between said central server and said proxy server; [[and]]

c) establishing communication between said proxy server and a client comprising a version manager adapted to generate a request for a <u>copy of a requested version of said file</u> ~~versioned file~~ from said repository through said proxy server to reduce network traffic between said central server and said client and to reduce load on said central server[[.]]<u>; and</u>

<u>d) causing said proxy server to provide said requested version from said read-only cache when available and by requesting said requested version from said central server otherwise.</u>

20. (previously presented) A computer-readable medium according to claim 19, further comprising instructions for the following steps:

maintaining a list of additional proxy servers in the version control system, said additional proxy servers being in addition to said proxy server in communication with said client; and

notifying said additional proxy servers in the list when a change is made to said versioned file.

21. (previously presented) A computer-readable medium according to claim 19, wherein said central server includes an access control system adapted to validate requests received by said central server from said proxy server, said computer-readable medium further comprising instructions for the following step: validating said request received by said central server.

22. (cancel)

23. (cancel)

24. (previously presented) A computer-readable medium according to claim 19, further comprising instructions for the following steps:

said client requesting via said proxy server modifications to said versioned file in said repository; and

said proxy server requesting said central server to make said modification to said versioned file in said repository.

25. (previously presented) A computer-readable medium according to claim 24, further comprising instructions for the following step:

controlling modification of the repository via a checkout mechanism comprising part of said central server.

26. (previously presented) A computer-readable medium according to claim 25, further comprising instructions for the following step:

keeping in the central server a log of changes made to the repository.

27. (previously presented) A computer-readable medium according to claim 26, further comprising instructions for the following step:

using the log to update the proxy server after a disruption to the communication between the proxy server and the central server.

28. (previously presented) A computer-readable medium according to claim 19, further comprising instructions for the following steps:

chaining together a plurality of proxy servers configured to serve a geographic area;

establishing communication between the plurality of chained together proxy servers and a regional one proxy server, said one proxy server being connected to said central server;

establishing communication between said one regional proxy server and said central server; and

sending updates from the central server for the plurality of chained together proxy servers first to the one regional proxy server.

29. (currently amended) A version control system according to claim 1, wherein said proxy server is adapted to operate transparently is transparent to users of said at least one client.